

THRIVE: Threshold Homomorphic encRyption based secure and privacy preserving biometric VErification system

Cagatay Karabat^{**†}, Mehmet Sabir Kiraz^{*}, Hakan Erdogan[†], Erkey Savas[†]

^{*}TUBITAK BILGEM UEKAE, Kocaeli, Turkey

[†]Sabanci University, TR-34956, Tuzla, Istanbul, Turkey

{cagatay.karabat, mehmet.kiraz}@tubitak.gov.tr, {haerdogan, erkays}@sabanciuniv.edu

Abstract—In this paper, we propose a new biometric verification and template protection system which we call the THRIVE system. The system includes novel enrollment and authentication protocols based on threshold homomorphic cryptosystem where the private key is shared between a user and the verifier. In the THRIVE system, only encrypted binary biometric templates are stored in the database and verification is performed via homomorphically randomized templates, thus, original templates are never revealed during the authentication stage. The THRIVE system is designed for the malicious model where the cheating party may arbitrarily deviate from the protocol specification. Since threshold homomorphic encryption scheme is used, a malicious database owner cannot perform decryption on encrypted templates of the users in the database. Therefore, security of the THRIVE system is enhanced using a two-factor authentication scheme involving the user's private key and the biometric data. We prove security and privacy preservation capability of the proposed system in the simulation-based model with no assumption. The proposed system is suitable for applications where the user does not want to reveal her biometrics to the verifier in plain form but she needs to proof her physical presence by using biometrics. The system can be used with any biometric modality and biometric feature extraction scheme whose output templates can be binarized. The overall connection time for the proposed THRIVE system is estimated to be 336 ms on average for 256-bit biohash vectors on a desktop PC running with quad-core 3.2 GHz CPUs at 10 Mbit/s up/down link connection speed. Consequently, the proposed system can be efficiently used in real life applications.

Index Terms—Biometric, Security, Privacy, Threshold Cryptography, Homomorphic Encryption, Malicious Attacks

I. INTRODUCTION

IN recent years, public and commercial organizations invest on secure electronic authentication (e-authentication) systems to reliably verify identity of individuals. Biometrics is one of the rapidly emerging technologies for e-authentication systems [1]. However, it is impossible to discuss biometrics without security and privacy issues [2], [3]. Biometrics, which are stored in a smart card or a central database, is under security and privacy risks due to increased number of attacks against identity management systems in recent years [2]–[5].

Security and privacy concerns on biometrics limit their widespread usage in real life applications. The initial solution that occurs to mind for security and privacy problems is to use cryptographic primitives. On the other hand, biometric templates cannot be directly used with conventional encryption

techniques (i.e. AES, 3DES) since biometric data are inherently noisy [6]. In other words, the user is not able to present exactly the same biometric data repeatedly. Namely, when a biometric template is encrypted during the enrollment stage, it should be decrypted to pass the authentication stage for comparison with the presented biometric. This, however, again leads to security and privacy issues for biometric templates at the authentication stage [6]. Another problem with regards to such a solution is the key management, i.e. storage of encryption keys. When a malicious database manager obtains decryption keys, he can perform decryption and obtain biometric templates of all users. Similar problems are valid for cryptographic hashing methods. Since cryptographic hash is a one-way function, when a single bit is changed the hash sum becomes completely different due to the avalanche effect [7]. Thus, successful authentication by exact matching cannot be performed even for legitimate users due to the noisy nature of biometric templates. Therefore, biometric templates cannot directly be used with the cryptographic hashing methods.

Biometric systems which use error correction methods are proposed to cope with noisy nature of the biometric templates in the literature [8]–[10]. In such systems, the biometric data collected at the enrollment stage is exactly the same with the biometric data collected at the authentication stage since they use error correction methods. In other words, these systems can get error-free biometric templates and thus cryptographic primitives (i.e. encryption and hashing) can successfully be employed without suffering from the avalanche effect [6], [10]–[12]. However, high error correcting capability requirements make them impractical for real life applications [13]. Furthermore, side information (parity bits) is needed for error correction and this may lead to information leakage and even other attacks (i.e., error correcting code statistics, and non-randomness attacks) [14]. Zhou *et al.* clearly demonstrate in their works that redundancy in an error correction code causes privacy leakage for biometric systems [15], [16].

Although biometric template protection methods are proposed to overcome security and privacy problems of biometrics [3], [17]–[30], recent research shows that security issues are still valid for these schemes [31]–[37]. Furthermore, there are a number of works on privacy leakages of biometric applications [38], [39], and yet more biometrics template protection methods [15], [16], [40]. In the literature, Zhou *et*

al. propose a framework for security and privacy assessment of biometric template protection methods [15]. In addition, Ignatenko *et al.* analyze the privacy leakage in terms of the mutual information between the public helper data and biometric features in a biometric template protection method. A trade-off between maximum secret key rate and privacy leakage is given in their works [39], [41].

Recently, homomorphic encryption methods are used with biometric feature extraction methods to perform verification via encrypted biometric templates [21], [42]–[44]. However, these methods offer solutions in the honest-but-curious model where each party is obliged to follow the protocol, but can arbitrarily analyze the knowledge that it learns during the execution of the protocol to obtain some additional information. The existing systems are not designed for the malicious model where each party can arbitrarily deviate from the protocol and may be corrupted. Moreover, they do not take into account security and privacy issues of biometric templates stored in the database [21], [44]. The authors state that their security model will be improved in the future work by applying encryption methods also on the biometric templates stored in the database. Furthermore, some of these systems are just designed for a single biometric modality or a specific feature extraction method which also limits their application areas [42], [43]. In addition, an adversary can enroll himself on behalf of any user to their systems since they do not offer any solutions for malicious enrollment. Finally, all these systems suffer from computational complexity.

Biohashing schemes are one of the emerging biometric template protection methods [26]–[30]. These schemes offer low error rates and fast verification at the authentication stage. However, they suffer from several attacks reported in the literature [34]–[37]. These schemes should be improved to be safely used in a wide range of real life applications. In our work, we develop new enrollment and authentication protocols for biometric verification methods. Our goal is to increase security and enhance privacy of the biometric schemes. The THRIVE system can work with any biometric feature extraction scheme whose outputs are binary or can be binarized. Since biohashing schemes can output binary templates called a biohash, they can be successfully used with the proposed system.

A. Our contributions

In this paper, we address adversary attacks in case of an active attacker who aims to gain access to the system in the malicious attack model. By taking these adversary attacks into account, we develop a new biometric authentication system based on threshold homomorphic cryptosystem. Our main goal is to increase the security of the system and preserve privacy of biometric templates of the users. The contributions of this work can be summarized as follows:

- A new biometric authentication system (which we call the THRIVE system) is proposed in the malicious model and the proposed system can be used with any existing biometric modality whose output can be binarized.
- Even if an adversary gains an access to the database and steals encrypted biometric templates, he can neither

authenticate himself by using these encrypted biometric templates due to the proposed authentication protocol, nor decrypt these encrypted biometric templates due to the $(2, 2)$ -threshold homomorphic encryption scheme.

- Only encrypted binary templates are stored in the database and biometric templates are never released even during authentication. Thus, the proposed system offers a new and advanced biometric template protection method without any helper data. In addition, only legitimate users can enroll in the proposed system since a signature scheme is used with the proposed enrollment protocol.
- The THRIVE system can be used in the applications where the user does not trust the verifier since she does not need to reveal her biometric template and/or private key to authenticate herself and the verifier does not need to reveal any data to the user with the proposed authentication protocol.
- Even if an adversary intercepts the communication channel between the user and the verifier, he cannot obtain any useful information on the biometric template since all exchanged messages are randomized and/or encrypted and he cannot perform decryption due to the $(2, 2)$ -threshold homomorphic encryption scheme. Furthermore, he cannot use the obtained data from message exchanges in this communication channel since nonce and signature schemes are used together in the authentication.
- The THRIVE system is a two-factor authentication system (biometric and secret key) and is secure against illegal authentication attempts. In other words, a malicious adversary cannot gain access to the proposed system without having the biometric data and the private key of a legitimate user by performing adversary attacks described in [45] as well as hill-climbing attacks [46]–[49].
- In the THRIVE system, the generated protected biometric templates are irreversible since templates are encrypted they are irreversible by definition as soon as decryption key is not stolen.
- The THRIVE system can generate a number of protected templates from the same biometric data of a user due to the randomized encryption and biohashing. Thus, it ensures diversity. Besides, they are also cancelable i.e. when they are stolen, they can be re-generated.
- A THRIVE authentication protocol run requires only 336 ms on average for 256 bit biohash vectors and 671 ms on average for 512 bit biohash vectors on a desktop PC with quad-core 3.2 GHz CPUs at 10 Mbit/s up/down link connection speed. Therefore, the proposed system is sufficiently efficient to be used in real-world applications.

The paper is structured as follows. Related work is addressed in Section 2. Preliminaries are described in Section 3. The proposed biometric authentication system is introduced in Section 4. Security proof of the proposed protocols are given in Section 5. Complexity analysis of the proposed system is discussed in Section 6. Section 7 concludes the paper.

II. RELATED WORK

Biometric template protection schemes are proposed to mitigate the security and privacy problems of biometrics [3], [18]–

[25]. However, various vulnerabilities of these technologies are reported in the literature [31]–[33]. Jain *et al.* classify biometric template protection schemes into two main categories [3]: 1) Feature transformation based schemes, 2) Biometric cryptosystems as illustrated in Figure ??.

The main idea behind biometric cryptosystems (also known as biometric encryption systems) is either binding a cryptographic key with a biometric template or generating the cryptographic key directly from the biometric template [50]. Thus, the biometric cryptosystems can be classified into two main categories: 1) Key binding schemes, 2) Key generation schemes. The biometric cryptosystems use helper data, which is public information, about the biometric template for verification. Although helper data are supposed not to leak any critical information about the biometric template, Rathgeb *et al.* show that helper data is vulnerable to statistical attacks [51]. Furthermore, Ignatenko *et al.* show how to compute a bound on possible secret rate and privacy leakage rate for helper data schemes [52]. Adler performs hill-climbing attack against biometric encryption systems [32]. In addition, Stoianov *et al.* propose several attacks (i.e., nearest impostors, error correcting code statistics, and non-randomness attacks) to biometric encryption systems [14].

In the literature, fuzzy commitment [10] and fuzzy vault schemes [25] are categorized under the key binding schemes. These schemes aim to bind a cryptographic key with a biometric template. In ideal conditions, it is infeasible to recover either the biometric template or the random bit string without any knowledge of the user's biometric data. However, this is not the case in reality because biometric templates are not uniformly random. Furthermore, error correction codes (ECC) used in biometric cryptosystems lead to statistical attacks (i.e., running ECC in a soft decoding or erasure mode and ECC Histogram attack) [14], [53]. Ignatenko *et al.* show that fuzzy commitment schemes leak information in cryptographic keys and biometric templates which lead to security flaws and privacy concerns [39], [41]. In addition, Zhou *et al.* argue that fuzzy commitment schemes leak private data. Chang *et al.* describe a non-randomness attack against fuzzy vault scheme which causes distinction between the minutiae points and the chaff points [54]. Moreover, Kholmatov *et al.* perform a correlation attack against fuzzy vault schemes [55].

Keys are generated from helper data and a given biometric template in key generation schemes [3]. Fuzzy key extraction schemes are classified under key generation schemes and use helper data [56]–[60]. These schemes can be used as an authentication mechanism where a user is verified via her own biometric template as a key. Although fuzzy key extraction schemes provide key generation from biometric templates, repeatability of the generated key (in other words stability) and the randomness of the generated keys (in other words entropy) are two major problems of them [3]. Boyen *et al.* describe several vulnerabilities (i.e. improper fuzzy sketch constructions may lead information on the secret, biased codes may cause majority vote attack, and permutation leaks) of the fuzzy key extraction schemes from outsider and insider attacker perspectives [61]. Moreover, Li *et al.* mention that when an adversary obtains sketches, they may reveal the

identity of the users [62].

Biohashing schemes are simple yet powerful biometric template protection methods [26]–[30] which can be classified under salting based schemes. It is worth pointing out that biohashing is completely different from cryptographic hashing. Although biohashing schemes are proposed to solve security and privacy issues, there are still security and privacy issues associated with them [34]–[37]. In these works, the authors claim that biohashes can be reversible under certain conditions and an adversary can estimate biometric template of a user from her biohash. Consequently, when biohashes are stored in the databases and/or smart cards in their plain form, they can threaten the security of the system as well as the privacy of the users. Moreover, an adversary can use an obtained biohash to threaten the system security by performing malicious authentication. Furthermore, when the secret key is compromised, an adversary can recover the biometric template since these schemes are generally invertible [3].

Non-invertible transform based schemes use a non-invertible transformation function, which is a one-way function, to make the biometric template secure [63]–[65]. User's secret key determines the parameters of non-invertible transformation function and this secret key should be provided at the authentication stage. Even if an adversary obtains the secret key and/or the transformed biometric template, it is computationally hard to recover the original biometric template. On the other hand, these schemes suffer from the trade-off between discriminability and non-invertibility which limits their recognition performance [3].

Another security approach is the use of cryptographic primitives (i.e. encryption, hashing) to protect biometric templates. These works generally focus on fingerprint-based biometric systems. Tuyls *et al.* propose the fingerprint authentication system which incorporates cryptographic hashes [66]. They use an error correction scheme to get exactly the same biometric template from the same user in each session which is similar to the fuzzy key extraction schemes. They store cryptographic hashes of biometric templates in the database and make comparison in the hash domain. However, there is no guarantee to get exactly the same biometric templates from the user even if the system incorporates an error correction scheme in real life applications since it is limited with the pre-defined threshold of error correction capacity. They also use helper data which are sent over a public channel and this may lead to security flaws as well. Moreover, an adversary can threaten the security of the system when he performs an attack against the database since he can obtain the user id, helper data and the hashed version of the secret which is generated by the biometric data and the helper data. Although the adversary cannot obtain the biometric data itself in its plain form, he can get all needed credentials (i.e. hash values of the secrets) to gain access to the system.

Kerschbaum *et al.* propose a protocol to compare fingerprint templates without actually exchanging them by using secure multi-party computation in the honest-but-curious model [67]. At the enrollment stage, the user gives her fingerprint template, minutiae pairs and PIN to the system. Thus, the verifier knows the fingerprint templates which are collected at the enrollment

stage. Although the user does not send her biometric data at the authentication, the verifier already has the user's enrolled biometric data and this threatens the privacy of the user in case of a malicious verifier. In addition, a malicious verifier can use these fingerprint templates for malicious authentication. Furthermore, since the fingerprint comparison reveals the matching scores (i.e. Hamming distance [68]), the attacker can perform a hill climbing attack against this system. Apart from these security and privacy flaws, the authors just focus on secure comparison in their protocol and they do not develop any solutions for the malicious model.

Erkin *et al.* [42] propose a privacy preserving face recognition system for the eigen-face recognition algorithm [69]. They design a protocol that performs operations on encrypted images by using the Pailler homomorphic encryption scheme. Later, Sadeghi *et al.* improve the efficiency of this system [43]. In both works, they use the eigen-face recognition algorithm together with homomorphic encryption schemes. However, they limit the recognition performance of the system with the eigen-face method although there are various feature extraction methods which perform better than it. Unfortunately, their system cannot be used for any other feature extraction method for face images. Moreover, they do not use a threshold cryptosystem which prevents from a malicious party aiming to perform decryption by himself. Storing face images (or corresponding feature vectors) in the database in plain is the most serious security flaw of this system. An adversary, who gains access to the database, can obtain all face images. Therefore, the adversary can perform the sixth attack type - attack against the database which definitely threatens the security of the system and the privacy of the users.

Barni *et al.* [21], [44] propose a privacy preservation system for fingercode templates by using homomorphic encryption in the honest-but-curious model. They, however, do not propose any security and privacy solutions on the biometric templates stored in the database. This issue is mentioned as a future work in their paper. In addition, they do not use threshold encryption which would prevent from a malicious party aiming to perform decryption by himself. Therefore, their proposed system is open to adversary attacks against the database as stated in their work. They do not address the malicious enrollment issue as well. Moreover, the user must trust the server in their system. Although they achieve better performance than [42], [43] in terms of bandwidth saving and time efficiency, they do not address the applications where the user and the verifier do not trust each other (e.g. the malicious model).

There are also some works on secure Hamming distance calculation by using cryptographic primitives [70]–[73]. These papers, however, limit their works only with secure Hamming distance calculation. These methods do not address biometric authentication as a whole and fails to satisfy security, privacy, template protection at the same time by taking into account computational efficiency which is very critical for real-world applications. Osadchy *et al.* [70] propose Pailler homomorphic encryption based secure Hamming distance calculation for face biometrics. The system is called SCiFI. Although they claim that SCiFI is computationally efficient, it mostly uses pre-computation techniques. Its pre-computation time includes

processing time that must be done locally by each user before using the system each time. They report that SCiFI's online running time takes 0.31 seconds for a face vector of size 900 bits however its offline computation time takes 213 seconds. Since these computations should be done by the user just before each attempt to use the system, the protocol is not that much efficient. Besides, SCiFI is only secure for semi-honest adversaries. Rane *et al.* [71] also propose secure Hamming distance calculation for biometric applications. However, their proposed method fails to ensure biometric database security since biometric templates are stored in plain format in the database. Thus, a malicious verifier can threaten a user's security and privacy. Bringer *et al.* [72] propose a secure Hamming distance calculation for biometric application. The system is called SHADE and it is based on committed oblivious transfer [74]. However, they also cannot guarantee biometric database security since biometric templates are stored in plain form in the database. Kulkarni *et al.* [73] propose a biometric authentication system based on *somewhat* homomorphic encryption scheme of Boneh *et al.* [75] which allows an arbitrary number of addition of ciphertexts but supports only one multiplication operation between the ciphertexts. Although the values stored on the enrollment server are the XORed values of the biometric template vector with the corresponding user's key, the user first extracts and sends her biometric features to the trusted enrollment server. Again this system uses a trusted enrollment server and fails to protect security and to preserve privacy of a user against a malicious database manager. In addition, the system is not efficient since 58 sec are required for successful authentication of a 2048 bit binary feature vector.

III. PRELIMINARIES

A. Threshold Homomorphic Cryptosystem

In this section, we briefly describe underlying cryptographic primitives of the protocols. Given a public key encryption scheme, let $m \in \mathcal{M}$ denote its message or plaintext space, $c \in \mathcal{C}$ the ciphertext space, and $r \in \mathcal{R}$ its randomness. Let $c = \text{Enc}_{pk}(m; r)$ depict an encryption of m under the public key pk where r is a random value. Let sk be its corresponding private key, which allows the holder to retrieve a message from a ciphertext. The decryption is done with the private key sk as $m = \text{Dec}_{sk}(c)$.

In a (t, n) -threshold cryptosystem, the knowledge of a private key is distributed among parties P_1, \dots, P_n . Then, at least t of these parties are required for successful decryption. On the other hand, there is a public key to perform encryption. More formally, let P_1, \dots, P_n be the participants. We define a (t, n) -threshold encryption scheme with three phases as follows:

- In the **key generation** phase, each participant P_i receives a pair (pk_i, sk_i) , where pk_i and sk_i are the *shares* of the public and secret key, respectively. Then, the overall public key pk is constructed by collaboratively *combining* the shares. Finally pk is broadcast to allow anyone to encrypt messages in \mathcal{M} . The shares of this public key are also broadcast which allow all parties to check the correctness of the decryption process.

- The **encryption** phase is done as in any public key encryption cryptosystem. If $m \in \mathcal{M}$ is the message, a (secret) random value r from \mathcal{R} is chosen and $c = \text{Enc}_{pk}(m; r)$ is broadcast under a public key pk .
- In the **threshold decryption** phase, given that t (or more) participants agree to decrypt a ciphertext c , they follow two steps. First, each participant produces a decryption share by performing $S_i^j = \text{Dec}_{sk_i^j}(c)$, $j = 1, \dots, t$. After broadcasting S_i^j , they all can apply a reconstruction function \mathcal{F} on these shares so that they can recover the original message by performing $m = \mathcal{F}(S_1^1, \dots, S_t^t)$ where P_1^1, \dots, P_t^t represent the group of t participants willing to recover m .

In case of a (t, n) -threshold scheme, the additional requirement is that if less than t parties gather their correct shares of the decryption of a given ciphertext, they will get no information whatsoever about the plaintext. In the proposed system, we use the $(2, 2)$ -threshold cryptosystem between the claimer (the user) and the verifier where both players must cooperate to decrypt.

A public key encryption scheme is said to be additively homomorphic if given $c_1 = \text{Enc}(m_1; r_1)$ and $c_2 = \text{Enc}(m_2; r_2)$ it follows that $c_1 c_2 = \text{Enc}(m_1 + m_2; r_3)$ where $m_1, m_2 \in \mathcal{M}$ and $r_1, r_2, r_3 \in \mathcal{R}$. There are various versions of threshold homomorphic cryptosystems. The most widely used are ElGamal [76] or Paillier [77] cryptosystems. In our proposal, we will use a threshold version of Goldwasser-Micali (GM) encryption scheme (i.e., between a user and a verifier) proposed by Katz and Yung in [78]. Note that GM scheme is XOR-homomorphic [79], i.e., given any two bits b_1, b_2 in $\{0, 1\}$, any random values $r_1, r_2 \in \mathcal{R}$, and any encryptions $\text{Enc}(b_1, r_1), \text{Enc}(b_2, r_2)$, it is easy to compute $\text{Enc}(b_1 \oplus b_2, r_1 r_2)$.

In the proposed protocol, we use a variant of the threshold decryption protocol which is the so-called private threshold decryption [80]. The requirement of this protocol is that one of the t parties will be the only party who will recover the secret. All $t - 1$ other parties follow the protocol and broadcast their shares to achieve this requirement. The party who will learn the plaintext proceeds with the decryption process privately, collects all decryption shares from the $t - 1$ other parties, and privately reconstructs the message. The remaining parties will not get any information about this message.

1) *Threshold XOR-Homomorphic Goldwasser-Micali Encryption Scheme:* We next give a brief explanation of $(2, 2)$ -GM cryptosystem between two users (in our proposal, between a user and a verifier) using a Trusted Dealer. We note that one can also exclude a trusted dealer using the scheme in [78]:

Key generation:

The trusted dealer first chooses prime numbers p and q ($\|p\| = \|q\| = n$) such that $N = pq$ and $p \equiv q \equiv 3 \pmod{4}$. The dealer next chooses $p_1, q_1, p_2, q_2 \in_R (0, 2^{2n})$ such that $p_1 \equiv q_1 \equiv 0 \pmod{4}$ and $p_2 \equiv q_2 \equiv 0 \pmod{4}$. He sets $p_0 = p - p_1 - p_2$ and $q_0 = q - q_1 - q_2$ and sends (p_1, q_1) to the first party and (p_2, q_2) to the second party. He finally broadcasts (p_0, q_0, N) .

Encryption of a bit $b \in \{0, 1\}$:

Choose $r \in_R \mathbb{Z}_N$ and compute a ciphertext $C = (-1)^{br^2}$

mod N .

Decryption:

All parties compute the Jacobi symbol $J = (\frac{C}{N})$. If $J \neq 1$ then all parties stop because either the encryption algorithm was not run honestly or the ciphertext was corrupted during the transmission. (Note that $(\frac{C}{N})$ is always 1, because $(\frac{C}{N}) = (\frac{C}{p})(\frac{C}{q}) = 1$ (i.e., either $(\frac{C}{p}) = 1$ and $(\frac{C}{q}) = 1$ or $(\frac{C}{p}) = -1$ and $(\frac{C}{q}) = -1$). If $J = 1$ then the first party broadcasts $b_1 = C^{(-p_1 - q_1)/4} \pmod{N}$. The second party (who is going to decrypt) will privately compute $b_0 = C^{(N - p_0 - q_0 + 1)/4} \pmod{N}$ and $b_2 = C^{(-p_2 - q_2)/4} \pmod{N}$. Finally, the decrypted bit b is computed as $b = (1 - b_0 b_1 b_2 \pmod{N})/2$.

Note that it is easy to see whether C is a quadratic residue by computing $b \equiv C^{(N - p - q + 1)/4} \pmod{N}$. The reason is briefly as follows. We first note that by Euler's theorem $C^{\phi(N)} \equiv 1 \pmod{N}$ where $\phi(N) = (p - 1)(q - 1)$. We also know that C is quadratic residue iff $C^{\phi(N)/2} \equiv 1 \pmod{N}$. If the Jacobi symbol $J = (\frac{C}{N}) = 1$ then by using $(\frac{C}{p})(\frac{C}{q}) = 1$ we have either $(\frac{C}{p}) = 1$ and $(\frac{C}{q}) = 1$ or $(\frac{C}{p}) = -1$ and $(\frac{C}{q}) = -1$. If $(\frac{C}{p}) = 1$ (resp. -1) and $(\frac{C}{q}) = 1$ (resp. -1) then $C^{p-1/2} \equiv 1 \pmod{p}$ (resp. $-1 \pmod{p}$) and $C^{q-1/2} \equiv 1 \pmod{q}$ (resp. $-1 \pmod{q}$). Hence, for both cases $C^{(p-1)(q-1)/4} \equiv 1 \pmod{p}$ and $C^{(p-1)(q-1)/4} \equiv 1 \pmod{q}$. By the Chinese Remainder Theorem, we have $C^{(p-1)(q-1)/4} \equiv 1 \pmod{N}$. Hence, C is quadratic residue iff $b = 1$.

B. Biometric Verification Scheme

Biometric verification schemes perform an automatic verification of a user based on her specific biometric data (e.g., face, fingerprint, iris). They have two main stages: 1) Enrollment stage, and 2) Authentication stage. The user is enrolled to the system at the enrollment stage. Then, she again provides her biometric data to the system at the authentication stage to prove her identity. Any biometric scheme, which provides binary outputs or whose outputs can be binarized, can work with the proposed threshold homomorphic cryptosystem. The THRIVE system can work with any biometric feature extraction method which produces fixed size vectors as templates and perform verification with distance calculations between the enrolled and the provided template at the authentication stage. When the output of a biometric feature extraction method is not binary, locality sensitive hashing can be used to binarize the feature vector [81]. After binarization, the binary templates can successfully be used with the proposed system. In this paper, we use biohashing as an example algorithm for extracting binary biometric templates. Although biohashing has its own security and privacy preservation mechanism, we do not rely on these for the security or the privacy preservation features. Thus it can be replaced with any other binary feature extraction method.

Biohashing schemes are simple yet powerful biometric template protection methods [26]–[30]. Biohash is a binary and pseudo-random representation of a biometric template. Biohashing schemes use two inputs: 1) Biometric template, 2) User's secret key. A biometric feature vector is transformed into a lower dimension sub-space using a pseudo-random set of orthogonal vectors which are generated from the user's

secret key. Then, the result is binarized to produce a pseudo-random bit-string which is called the biohash. In an ideal case, the Hamming distance between the biohashes belonging to the biometric templates of the same user is expected to be relatively small. On the other hand, the distance between the biohashes belonging to different users is expected to be sufficiently high to achieve higher recognition rates.

We describe the random projection (RP) based biohashing scheme proposed by Ngo *et al.* [82]. In this scheme, there are three main steps: 1) Feature extraction, 2) Random projection, 3) Quantization. These steps are explained for face biometrics.

1) *Feature Extraction*: The feature extraction is performed on the face images, which are collected at the enrollment stage, belonging to the users, $\mathbf{I}_{i,j} \in \mathbb{R}^{m \times n}$ where $i = 1, \dots, n$ and n denotes number of users, $j = 1, \dots, L$ and L denotes number of training images per user. The face images are lexicographically re-ordered and the training face vectors, $\mathbf{x}_{i,j} \in \mathbb{R}^{(mn) \times 1}$, are obtained. Then, Principle Component Analysis (PCA) [69] is applied.

$$\mathbf{y}_{i,j} = \mathbf{A}(\mathbf{x}_{i,j} - \mathbf{w}) \quad (1)$$

where $\mathbf{A} \in \mathbb{R}^{k \times (mn)}$ is the PCA matrix trained by the face images in the training set, \mathbf{w} is the mean face vector, and $\mathbf{y}_{i,j} \in \mathbb{R}^{k \times 1}$ is vector containing PCA coefficients belonging to the j^{th} training image of the i^{th} user.

2) *Random Projection*: At this phase, a RP matrix, $\mathbf{R} \in \mathbb{R}^{\ell \times k}$, is generated to reduce the dimension of the PCA coefficient vectors. The RP matrix elements are independent and identically distributed (*i.i.d*) and generated from a Gauss distribution with zero mean and unit variance by using a Random Number Generator (RNG) with a seed derived from the user's secret key. The Gram-Schmidt (GS) procedure is applied to obtain an orthonormal projection matrix $\mathbf{R}_{GS} \in \mathbb{R}^{\ell \times k}$ to have more distinct projections. Finally, PCA coefficients are projected onto a lower ℓ -dimensional subspace.

$$\mathbf{z}_{i,j} = \mathbf{R}_{GS} \mathbf{y}_{i,j} \quad (2)$$

where $\mathbf{z}_{i,j} \in \mathbb{R}^{\ell \times 1}$ is an intermediate biohash vector belonging to the j^{th} training image of the i^{th} user.

3) *Quantization*: At this phase, the intermediate biohash vector $\mathbf{z}_{i,j}$ elements are binarized with respect to the threshold.

$$\lambda_{i,j}^k = \begin{cases} 1 & \text{if } z_{i,j}^k \geq \beta \\ 0 & \text{Otherwise.} \end{cases} \quad (3)$$

where $\lambda_{i,j} \in \{0, 1\}^\ell$ denotes biohash vector of the j^{th} training image of the i^{th} user and β denotes the mean value of the intermediate biohash vector $\mathbf{z}_{i,j}$.

A biohash vector, \mathbf{B}_{enroll_i} , for the i^{th} user is stored in the database at the enrollment stage for verification purpose during the authentication stage. Note that, \mathbf{B}_{enroll_i} can be any vector among $\lambda_{i,j}$ vectors in a real-world application. For simulation purposes, we take into account all possible biohashes for a user by computing $\lambda_{i,j}$. The user is authenticated if the Hamming distance between \mathbf{B}_{enroll_i} and \mathbf{B}_{auth_i} is below a threshold μ .

$$\sum_{k=1}^n \mathbf{B}_{enroll_i}^k \oplus \mathbf{B}_{auth_i}^k \leq \mu \quad (4)$$

where $\mathbf{B}_{enroll_i}^k$ denotes the k^{th} bit of \mathbf{B}_{enroll_i} , $\mathbf{B}_{auth_i}^k$ denotes the k^{th} bit of \mathbf{B}_{auth_i} , and \oplus denotes the binary XOR (exclusive OR) operator. Consequently, the verifier decides whether the claimer is a legitimate user or not according the threshold.

IV. THE PROPOSED BIOMETRIC AUTHENTICATION SYSTEM

In this section, the proposed biometric authentication system is introduced. In the proposed system, there are two major roles: 1. *User* (U_i) and 2. *Verifier* (V). The user has control of the biometric sensor, the feature extractor, and the biohash generator whereas the verifier has control of the database and the matcher. We assume that there is a trusted third party (TTP) which initially sets up the system public/private keys.

The TTP distributes the keys in the proposed system. There are public-private key pairs $(pk_i, (sk_i^1, sk_i^2))$ which are shared between the user and the verifier. pk_i is the public key of the i^{th} user, U_i , and both the user and the verifier have it. Recall that, when an enrollment biometric template is encrypted by pk_i , this can solely be decrypted using the private key shares of the user (sk_i^1) and the verifier (sk_i^2) collaboratively since the proposed system is based on the $(2, 2)$ -threshold homomorphic cryptosystem. Here, sk_i^1 is the private key share of the i^{th} user, U_i , and sk_i^2 is the private key share of the verifier. Besides, there is a public-private key pair (pk_{U_i}, sk_{U_i}) which belongs to the i^{th} user, U_i , where pk_{U_i} is the public key and sk_{U_i} is its associated private key to perform the signature operation. The verifier also has the public key pk_{U_i} of the i^{th} user, U_i .

A. Enrollment Stage

At this stage, the i^{th} user U_i has control over the biometric sensor, the feature extractor, and the biohash generator whereas the verifier has control over the database as illustrated in Figure 1. It is worth mentioning that biometric sensor authentication must be achieved in the proposed system before executing the enrollment protocol to prevent unauthorized sensors to be used as clients in the system by malicious users. This, however, is not explicitly indicated in the protocol in order not to clutter the paper. The proposed enrollment protocol is illustrated in Figure 2 and steps of it are introduced as follows:

- 1) **Step 1**: The i^{th} user, U_i , computes her biohash, $\mathbf{B}_{enroll_i} = B_{enroll_i}^1 \dots B_{enroll_i}^n$ where $B_{enroll_i}^j \in \{0, 1\}$, $j = 1, \dots, n$. Next, the user encrypts her biohash, $C_i^j = \text{Enc}_{pk_i}(B_{enroll_i}^j) \forall j = 1, \dots, n$, by using the public key pk_i . Then, the user signs her encrypted biohash, $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$, and sends it to the verifier.
- 2) **Step 2**: The verifier V verifies $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$ by using pk_{U_i} and stores the signature and encrypted biohash in the database. These data will be used for verification at the authentication stage.

Note that the proposed enrollment protocol uses the $(2, 2)$ -threshold homomorphic cryptosystem. Namely, both the user and the verifier have to cooperate to decrypt a ciphertext. Furthermore, the signature ensures that the data stored in the database are generated by a legitimate user.

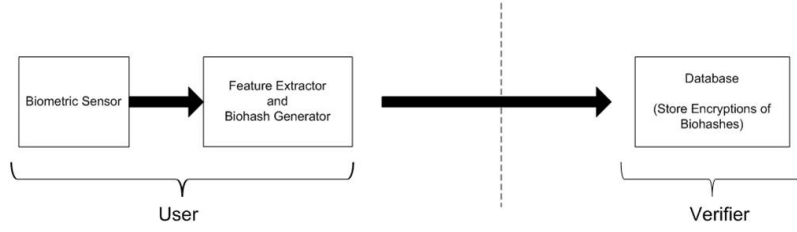


Fig. 1. Illustration of the THRIVE enrollment stage: the user has control over the biometric sensor, the feature extractor and the biohash generator whereas the verifier has control over the database.

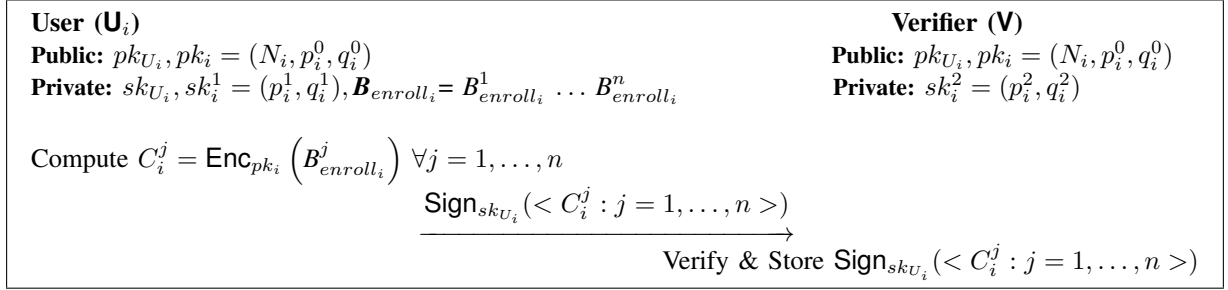


Fig. 2. The Proposed Enrollment Protocol

Lemma 1. *Biohashes are not revealed at the enrollment stage.*

Proof. (Sketch) At the enrollment stage, the i^{th} user U_i first encrypts her biohash and then signs it. After these computations, U_i sends her encrypted and signed biohash $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$ to the verifier. Since the user's biohash is not sent in plain form, biohashes are not revealed to the verifier at the enrollment stage.

Lemma 2. *An adversary cannot register as a legitimate user at the enrollment stage.*

Proof. (Sketch) At the enrollment stage, the i^{th} user U_i encrypts her biohash by using the public key pk_i and then signs her encrypted biohash by using her private key sk_{U_i} . Thus, U_i sends encrypted and signed biohash $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$ to the verifier. The verifier knows pk_{U_i} of the users. Since the verifier verifies the signature of the user, an adversary cannot register himself as a genuine user without having the private key of her sk_{U_i} for computing $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$.

B. Authentication Stage

At this stage, the i^{th} user U_i , has control over the biometric sensor, the feature extractor, and the biohash generator whereas the verifier has control over the database, the matcher and the decision maker as illustrated in Figure 4. U_i tries to prove herself to the verifier by executing the proposed authentication protocol shown in Figure 3. Similar to the enrollment case, the biometric sensor must be authorized by the system before the authentication protocol is carried out. Steps of the proposed authentication protocol are as introduced as follows:

- 1) **Step 1:** U_i wants to verify her identity by using her biohash and sends a connection request to the verifier. Then, U_i computes her biohash $\mathbf{B}_{auth_i} = B_{auth_i}^1 \dots B_{auth_i}^n$ where $B_{auth_i}^j \in \{0, 1\}, j = 1, \dots, n$. Note

that the user cannot produce exactly the same biometric template at each attempt and this results in different biohashes computed by the same user. Therefore, \mathbf{B}_{enroll_i} and \mathbf{B}_{auth_i} are different biohashes although they are generated by the same user at different sessions (enrollment and authentication). First, U_i chooses a random vector $\mathbf{r}_i^j \in_R \{0, 1\} \forall j = 1, \dots, n$. She computes $R_i^j = r_i^j \oplus B_{auth_i}^j \forall j = 1, \dots, n$. Then, U_i generates a nonce, nonce_{U_i} , which is uniquely defined and contains information about user id, session id and timestamp. Finally, the user sends $< R_i^j : j = 1, \dots, n >$, nonce_{U_i} to the verifier.

- 2) **Step 2:** The verifier retrieves $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$ from the database where $C_i^j = \text{Enc}_{pk_i}(B_{enroll_i}^j) \forall j = 1, \dots, n$. Then, it generates a nonce nonce_{V_i} which contains information about the verifier, session id and timestamp. Finally, it sends $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$, nonce_{V_i} to the user.
- 3) **Step 3:** The user verifies $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$ by using public key pk_{U_i} . She computes $C_i'^j = \text{Enc}_{pk_i}(r_i^j) \cdot C_i^j = \text{Enc}_{pk_i}(r_i^j \oplus B_{enroll_i}^j) \forall j = 1, \dots, n$. Then, she performs partial decryption over $C_i'^j$, i.e., $T_i^{1,j} = \text{Dec}_{sk_i^1}(C_i'^j) = (C_i'^j)^{(-p_i^1 - q_i^1)/4} \mod N_i, \forall j = 1, \dots, n$ using her private key share sk_i^1 . Finally, she sends $\text{Sign}_{sk_{U_i}}(< \text{Enc}_{pk_i}(r_i^j), T_i^{1,j} : j = 1, \dots, n >)$, nonce_{U_i} , nonce_{V_i} to the verifier.
- 4) **Step 4:** V verifies the signature $\text{Sign}_{sk_{U_i}}(< \text{Enc}_{pk_i}(r_i^j), T_i^{1,j} : j = 1, \dots, n >)$, nonce_{U_i} , nonce_{V_i} by using the public key pk_{U_i} . Then, it computes $C_i''^j = \text{Enc}_{pk_i}(r_i^j) \cdot C_i^j$ (this is done to assure correctness of the result and will prevent a malicious user computing different values than expected.). Next, the verifier performs the full decryption by computing $T_i^{2,j} = \text{Dec}_{sk_i^2}(C_i''^j) = (C_i''^j)^{(-p_i^2 - q_i^2)/4} \mod N_i$

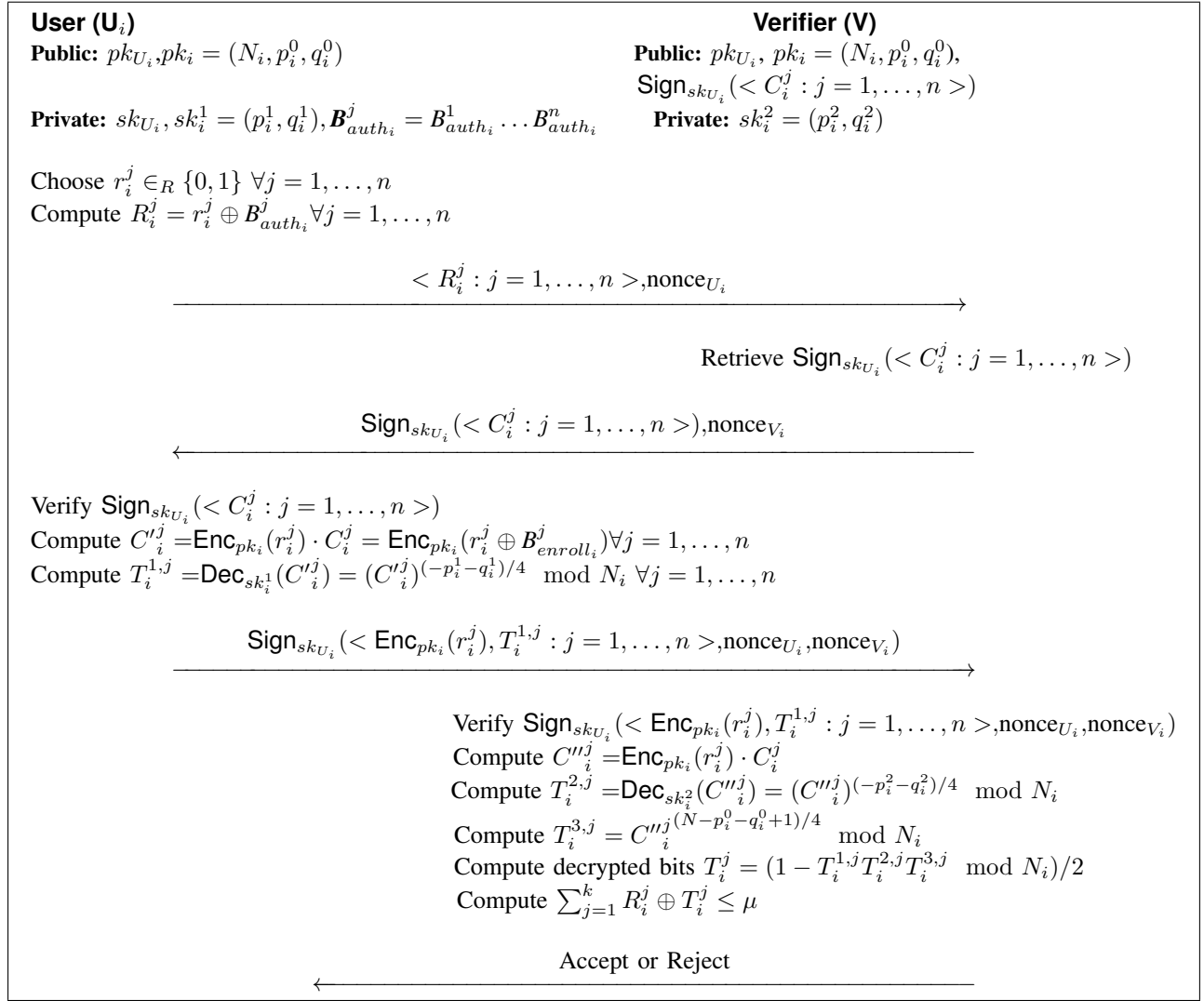


Fig. 3. The Proposed Authentication Protocol

and $T_i^{3,j} = (C_i''^j)^{(N - p_i^0 - q_i^0 + 1)/4} \mod N_i$. Finally, the verifier computes the decrypted j^{th} bits $T_i^j = (1 - T_i^{1,j} T_i^{2,j} T_i^{3,j} \mod N_i) / 2$ and the Hamming distance between R_i^j and T_i^j is calculated as follows:

$$\sum_{j=1}^n R_i^j \oplus T_i^j \leq \mu \quad (5)$$

where μ is the distance threshold. Therefore, the verifier decides whether the user is authentic with respect to the pre-defined distance threshold. Note that the Hamming distance between $r_i^j \oplus B_{enroll_i}^j$ and $r_i^j \oplus B_{auth_i}^j$ is equal to the Hamming distance between $B_{enroll_i}^j$ and $B_{auth_i}^j$. Finally, the verifier sends its decision (either Accept or Reject) to the user. However, the user may get dummy output if there is an error or an attack (i.e., override response attack) in the communication channel. The proposed system can easily be updated to cope with such an attack, for instance, by allowing the verifier to sign its decision including the nonces generated during the authentication session (i.e., either $\text{Sign}(\text{Accept}, nonce_{U_i},$

$nonce_{V_i})$ or $\text{Sign}(\text{Reject}, nonce_{U_i}, nonce_{V_i})$ and then sends it to the user. In this way, authenticity, integrity and origin of the data can easily be verified. Besides, signing the nonces ($nonce_{U_i}$ and $nonce_{V_i}$) also makes the communication unique and avoids replay attacks.

Lemma 3. Biohashes are not revealed at the authentication stage.

Proof. Authentication is performed in a randomized domain. In other words, the authentication is determined by comparing R_i^j and T_i^j . An adversary can only obtain R_i^j and T_i^j which are revealed at the authentication stage. Recall that these are randomized biohashes. Thus, from the adversary's perspective, there are three unknowns (r_i^j , $B_{enroll_i}^j$ and $B_{auth_i}^j$) and two equations which are shown in the below.

$$T_i^j = r_i^j \oplus B_{enroll_i}^j \quad (6)$$

$$R_i^j = r_i^j \oplus B_{auth_i}^j \quad (7)$$

where r_i^j is the random bit generated by the U_i for the j^{th} bit. Since this is a system of linear equations with fewer equations

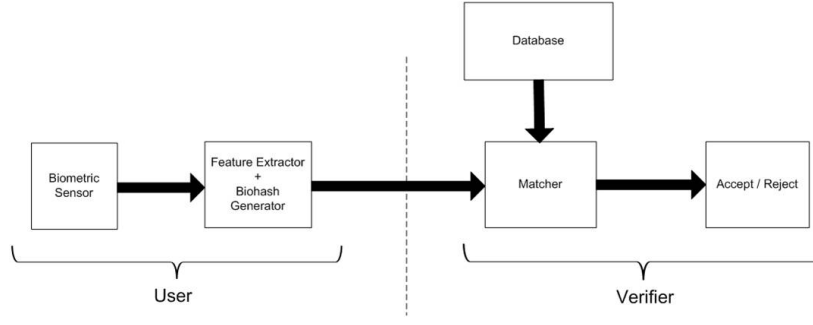


Fig. 4. Illustration of the THRIVE authentication stage: the user has control over the biometric sensor, the feature extractor and the biohash generator whereas the verifier has control over the database, the matcher and the decision maker.

than unknowns, the system has infinitely many solutions. Consequently, it is impossible for the adversary to obtain a legitimate user's biohash by using T_i^j and R_i^j which are revealed at the authentication stage. As a result, the proposed biometric authentication system ensures security and privacy.

V. SECURITY PROOF OF THE PROPOSED AUTHENTICATION PROTOCOL

In this section, we prove that the proposed authentication protocol shown in Figure 3 is secure against malicious user and verifier. In this proof, there exists a probabilistic polynomial-time simulator that produces a protocol transcript which is statistically indistinguishable from the one resulting from a real execution of the authentication protocol. The simulator must perform its task without knowing the private information of the party who proves her identity [83]. We show that given a party is corrupted (either a user or verifier), there exists a simulator that can produce a view which is statistically indistinguishable from the view of that party interacting with the other honest party. Assuming that one party is corrupted, we build an efficient simulator that has access to the public input and private secret shares of the secret key of the corrupted party. Besides, the simulator knows the public output. We want to point out that the simulator already knows the shares of the secret key of the corrupted party before the simulation is run. Since the threshold cryptosystem is set up before the protocol starts, we assume that the simulator extracts this information when the distributed key generation is run.

It is worth mentioning that the proposed authentication protocol gives computational privacy to both the user and the verifier due to the semantic security of the underlying cryptosystem. Furthermore, it is shown that the proposed authentication protocol is simulatable for both parties and these simulations produce views which are statistically indistinguishable from the views in the real protocol executions.

Theorem 1. *The proposed authentication protocol, which is shown in Figure 3, is secure in the presence of static malicious adversaries.*

Proof. We show that given a party is corrupted, there exists a simulator that can produce a view to the adversary that is statistically indistinguishable from the view in the real protocol execution based on its private decryption share as well as public information.

Case 1 - User U_i is corrupted. In this case, we prove the security for the case where U_i is corrupted. The simulator has the private key share of the user sk_i^1 , the user's private key sk_{U_i} , and the user's biohash $B_{auth_i}^j$ apart from the user's public information (i.e., pk_{U_i} and pk_i) as described in the proposed authentication protocol. The simulator constructs a view for the user which is statistically close to the one the user observes when interacting with the honest verifier by using this information. The simulator proceeds as follows:

- 1) The simulator first obtains $\langle R_i^j : j = 1, \dots, n \rangle, \text{nonce}_{U_i}$. As in the second round of the real protocol, the simulator needs to output the signature of the encrypted biohash of the user. To do so, the simulator computes $\tilde{C}_i^j = \text{Enc}_{pk_i}(B_{auth_i}^j) \forall j = 1, \dots, n$ by using the user's public key pk_i , and then computes $\text{Sign}_{sk_{U_i}}(\langle \tilde{C}_i^j : j = 1, \dots, n \rangle)$. The simulator also generates a nonce called nonce_{V_i} . The values $\text{Sign}_{sk_{U_i}}(\langle \tilde{C}_i^j : j = 1, \dots, n \rangle)$ and nonce_{V_i} are the simulated outputs. Note that the simulator uses $\tilde{\mathbf{B}}_{auth}$ instead of \mathbf{B}_{enroll} since it is the only available biohash to him.
- 2) The simulator obtains $\text{Sign}_{sk_{U_i}}(\langle \text{Enc}_{pk_i}(r_i^j), T_i^{1,j} : j = 1, \dots, n \rangle, \text{nonce}_{U_i}, \text{nonce}_{V_i})$ as in the second round of the protocol. The simulator next verifies the signature $\text{Sign}_{sk_{U_i}}(\langle \text{Enc}_{pk_i}(r_i^j), T_i^{1,j} : j = 1, \dots, n \rangle, \text{nonce}_{U_i}, \text{nonce}_{V_i})$ that U_i would run. Next, it computes $\tilde{C}_i''^j = \text{Enc}_{pk_i}(r_i^j) \cdot \tilde{C}_i^j$. Given $\tilde{C}_i''^j$, its plaintext and the share of private key sk_i^2 of the user U_i the decryption shares $T_i^{2,j}$ can be simulated as follows: The simulator computes $\tilde{b}_0 = [\tilde{C}_i''^j]^{(N-p_0-q_0+1)/4} \bmod N$ from the public information and computes $\tilde{b}_1 = [\tilde{C}_i''^j]^{(-p_1-q_1)/4} \bmod N$ since it knows sk_i^1 (i.e., p_1, q_1). Let's denote \tilde{b} for the plaintext of $\tilde{C}_i''^j$. Then, the simulator can compute $\tilde{b}_2 \bmod N \equiv (1 - 2\tilde{b})/(\tilde{b}_0\tilde{b}_1) \bmod N$ (which is $T_i^{2,j}$ in the real protocol). Note that in the real setting this is not possible since the plaintext inside the ciphertext is unknown $\tilde{C}_i''^j$. Similarly, $T_i^{3,j}$ can also be simulated since p_i^0 and q_i^0 are known by the simulator. The simulator finally computes $\sum_{j=1}^k R_i^j \oplus T_i^j$.

Each step of the proposed authentication protocol for the simulator is simulated and this completes the simulation for

the malicious user. The transcript is consistent and statistically indistinguishable from the user's view when interacting with the honest verifier.

Case 2 - The verifier V is corrupted. We now prove the security for the case where the verifier is corrupted. The simulator has the private key share of the verifier (sk_i^2) apart from the verifier's public information (i.e., pk_{U_i} , pk_i , and $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$) as described in the proposed authentication protocol. The simulator constructs a view for the verifier which is statistically close to the one when interacting with the honest user by using this information. The simulator proceeds as follows:

- 1) Note that the simulator already knows $< C_i^j : j = 1, \dots, n >$ because of the knowledge of $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$. The simulator chooses a random bit \tilde{r}_i^j and arbitrary $\tilde{B}_{auth_i}^j \in_R \{0, 1\}$ and computes $\tilde{R}_i^j = \tilde{r}_i^j \oplus \tilde{B}_{auth_i}^j$. Recall that the simulator must perform its task without knowing the private information of the honest user in this case. Thus, although it does not have real r_i^j and $B_{auth_i}^j$ it can successfully execute the simulated conversation since \tilde{R}_i^j is uniformly random.
- 2) The simulator obtains $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$. The simulator verifies the signature $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$ (by using the user's public key pk_{U_i}) that V would run. Next, it next computes $\tilde{C}_i'^j = \text{Enc}_{pk_i}(\tilde{r}_i^j) \cdot C_i^j$.
- 3) Given $\tilde{C}_i'^j$, its plaintext (which is $\tilde{r}_i^j \oplus B_{enroll_i}^j$) and the share of private key sk_i^2 of the verifier V the decryption share $\tilde{T}_i^{1,j}$ can also be simulated as follows: The simulator computes $\tilde{b}_0 = [\tilde{C}_i'^j]^{(N-p_0-q_0+1)/4} \bmod N$ from the public information and computes $\tilde{b}_2 = [\tilde{C}_i'^j]^{(-p_2-q_2)/4} \bmod N$ since it knows sk_i^2 (i.e., p_2, q_2). Let's denote \tilde{b} for the plaintext of $\tilde{C}_i'^j$. Then, the simulator can compute $\tilde{b}_1 \bmod N \equiv (1 - 2b)/(\tilde{b}_0\tilde{b}_2) \bmod N$ (which is $T_i^{1,j}$ in the real protocol). Note that in the real setting this is not possible since the plaintext inside the ciphertext is unknown $\tilde{C}_i''^j$.
- 4) Finally, the simulator needs to simulate the signature $\text{Sign}_{sk_{U_i}}(< \text{Enc}_{pk_i}(r_i^j), T_i^{1,j} : j = 1, \dots, n >, \text{nonce}_{U_i}, \text{nonce}_{V_i})$. However, this is not possible since the simulator does not know sk_{U_i} . In order to successfully simulate this final step, we need to provide additional information to the simulator. For example, performing a following modification over the the key generation phase in the real protocol the simulation will be possible:
 - During the key generation in the real protocol, the private key sk_{U_i} is distributed to the user U_i and V in a threshold fashion. For example, distributed RSA setting can be used for the signature algorithm (note that for an RSA setting (e, n) denotes the public key and (p, q, d) denotes the private key where $n = pq$ and $ed \equiv 1 \bmod (p-1)(q-1)$). Namely, the private key d can be divided into d_1 and d_2 such that the ciphertext c can be decrypted together with U_i and V as $m \equiv c^d \equiv c^{d_1+d_2} \bmod n$.
 - In order to simulate, instead of signing procedures, U_i will compute an encryption, compute its partial decryption and finally will send to the verifier. This

will assure that the user indeed used its private decryption key over the encrypted value. Next, the verifier will also compute its partial decryption and will compute the decrypted value privately.

Hence, with this modified version the decryption share can be simulated in a similar way as described at the third step of the simulation.

Consequently, each step of the proposed authentication protocol for the simulator is simulated and this completes the simulation for the malicious verifier. The transcript is consistent and statistically indistinguishable from the verifier's view when interacting with the honest user. \square

VI. COMPLEXITY ANALYSIS OF THE PROPOSED SYSTEM

In this section, we discuss the complexity of the THRIVE enrollment and authentication protocols. The complexity of the THRIVE enrollment and authentication protocols are examined in terms of protocol steps for the round complexity, the number of cryptographic operations for the computational complexity and the number of messages exchanged by the two parties for the communication complexity. Without loss of generality, we will provide complexity of the THRIVE protocols using the (2,2)-threshold homomorphic GM cryptosystem as an instance [79]. In the protocol we use (2,2)-threshold XOR homomorphic GM cryptosystem for confidentiality (i.e., encryption and decryption) while for signature generation and verification a conventional cryptosystem such as RSA (using the key pair (pk_{U_i}, sk_{U_i})) is employed.

The round complexity of the enrollment protocol is only one. For the computational complexity, the enrollment protocol requires n XOR-homomorphic encryptions, and one conventional signature generation for a user, but one signature verification for the server. For the communication complexity, the user sends a conventional signature and n ciphertexts (i.e., C_i^j for $j = 1, \dots, n$).

In the authentication protocol, there are only four rounds. For the computational complexity of the authentication protocol, the user generates one conventional signature and verifies another, computes n XOR-homomorphic encryptions and n XOR-homomorphic decryptions, and performs n modular multiplications over homomorphic ciphertexts (i.e., $\text{Enc}_{pk_i}(r_i^j) \cdot C_i^j$ for $j = 1, \dots, n$). The verifier verifies one conventional signature, computes $n + 2$ modular multiplications, $2n$ decryptions, and performs n Jacobi computations to check $\text{Enc}_{pk_i}(r_i^j)$ for $j = 1, \dots, n$. In total, there are n XOR-homomorphic encryptions, $3n$ XOR-homomorphic decryptions, two signature verifications, one signature generation, n Jacobi computations, and $2n + 2$ modular multiplications during the entire authentication protocol.

For the communication complexity of the authentication protocol, the user sends $2n$ homomorphic ciphertexts, one conventional signature and one nonce value. The verifier sends one conventional signature, n homomorphic ciphertexts, and one nonce to the user. In total, $3n$ homomorphic ciphertexts, two conventional signatures, and two nonce values are exchanged.

	Irreversibility	Cancellation	Diversity	Helper Data Usage	Database Security	Homomorphic Encryption	Threshold Homomorphic Encryption	Malicious Attack Model
THRIVE System	✓	✓	✓	✗	✓	✓	✓	✓
Salting Based Schemes	✗	✓	✓	✗	✗	✗	✗	✗
Non-Invertible Transform Based Schemes	✓	✓	✓	✗	✗	✗	✗	✗
Key Binding Schemes	✓	✓	✓	✓	✗	✗	✗	✗
Key Generation Schemes	✓	✓	✓	✓	✗	✗	✗	✗
Tuyil's System [70]	✓	✓	✓	✓	✗	✗	✗	✗
Kerschbaum's Systems [71]	✓	✓	✓	✗	✗	✓	✗	✗
Erkin's System [42]	✓	✓	✓	✗	✗	✓	✗	✗
Barni's System [21,44]	✓	✓	✓	✗	✗	✓	✗	✗
Osadchy's System (SciFi) [74]	✓	✓	✓	✗	✗	✓	✗	✗
Rane's System [75]	✓	✓	✓	✗	✗	✓	✗	✗
Bringer's System [76]	✓	✓	✓	✗	✗	✗	✗	✓
Kulkarni's System [77]	✓	✓	✓	✗	✗	✓	✗	✗

Fig. 5. Comparison between the THRIVE system and the existing solutions. ✓ denotes that the system satisfies the property whereas ✗ denotes that the system does not satisfies the property.

In the following, we provide timing estimates for the entire protocol for 80-bit security level, on a desktop computer, which has Intel processor with various clock speeds (2.4 GHz and 3.2 GHz). On the computer one modular multiplication using Montgomery arithmetic takes about 2000 clock cycles. Furthermore, one decryption operation in XOR homomorphic GM cryptosystem takes only a single modular multiplication. On the other hand, one decryption in XOR homomorphic GM cryptosystem requires one modular exponentiation operation which takes about 3.2 million clock cycles. In addition, one signature generation and verification operation in conventional cryptosystem such as RSA are equivalent to one modular exponentiation operation.

The bandwidth usage of the proposed protocol for various lengths of biohashes of the user are given in Table I. The required bandwidth for the proposed protocol increases with the increasing length of the biohashes. Bandwidth usage also affects the overall connection time.

TABLE I
BANDWIDTH (TOTAL NUMBER OF BITS EXCHANGED) USAGE OF THE PROPOSED PROTOCOL.

Length of Biohash	Bandwidth (Kbits)	Time @ 10 Mbit/s (ms)
112	348	35
192	594	59
256	791	79
512	1577	158
2048	6296	630

The computation times for the user and the verifier at 2.4 GHz for the proposed protocol with different biohash lengths are given in Table II. Naturally, it is expected that the required computation times of the proposed protocol increase as lengths of the biohashes increase.

We compare the communication complexity of the proposed system with the existing systems in the literature assuming that all systems run on a computer platform with 2.4 GHz clock speed. Erkin *et al.*'s system [42] requires 56.25 ms, Barni *et al.*'s system [21], [44] requires 50 ms and Sadeghi *et al.*'s system [43] requires 25 ms for authentication at the server side for single user with 112 bit binary feature vector

TABLE II
COMPUTATION TIME FOR THE USER AND THE VERIFIER AT 2.4 GHZ.

Length of Biohash	User Time (ms)	Verifier Time (ms)
112	151	449
192	258	769
256	343	1026
512	685	2050

[21]. On the other hand, our solution requires 449 ms for the same authentication set-up. Although existing solutions seem faster than the proposed system, they propose their solutions in the semi-honest model whereas our solution is secure under malicious adversary model. The comprehensive comparison between the proposed system and existing systems are shown in Figure 5. It is clearly seen in Figure 5 that the proposed THRIVE system offers superior security and privacy preservation solutions under malicious attack model.

The similar timing estimations are also computed for 3.2 GHz as shown in Table III.

TABLE III
COMPUTATION TIME FOR THE USER AND THE VERIFIER AT 3.2 GHZ.

Length of Biohash	User Time (ms)	Verifier Time (ms)
112	113	337
192	193	577
256	257	769
512	514	1537
2048	2051	6146

We compare the communication complexity of the proposed system with the existing systems at 3.2 GHz in the literature. Kulkarni *et al.*'s system [73] requires 58 s at the server side, 10 ms at the user side and 400 Kbit bandwidth usage for authentication of single user with 2048-bit binary feature vector at 3.2 GHz. Our proposed system requires 6146 ms at the server side, 2051 ms at the user side and 6296 Kbit bandwidth usage for the same authentication set-up. Thus, it is faster than Kulkarni *et al.*'s system. In addition, our proposes system offers other advantages such as that it is secure under malicious adversary model, that the biometric is protected via both a template protection method (e.g., biohash) and cryptographic primitives (e.g. threshold homomorphic encryption). On the other hand, Kulkarni *et al.*'s system offers solution for semi-honest model which also means that it is insecure for the malicious model. The comprehensive comparison between the proposed system and existing systems are shown in Figure 5.

VII. CONCLUSION

In this work, we propose a novel biometric authentication system. The aim of the proposed system is to increase security against adversary attacks defined in [45] when an adversary wants to gain access to the system as a legitimate user. We propose novel enrollment and authentication protocols to increase the security against attacks reported in the literature and preserve the privacy of users. The proposed system can be used with any biometric feature extraction method which can produce binary templates or whose outputs can be binarized. The biohashing is chosen as an example binary biometric template generation system since it offers satisfactory error

rates and fast authentication. The comparison is performed in a randomized domain in the authentication stage and the binary templates (e.g., biohashes) are never released. In addition, only encrypted binary templates are stored in the database. Since we use the (2,2)-threshold cryptosystem, the verifier cannot decrypt the data stored in the database by himself. Namely, the user and the verifier both has to cooperate to decrypt the encrypted binary templates. The proposed system can be used in applications where the user is not willing to reveal her biometrics to the verifier although she needs to proof her physical presence by using biometrics. The THRIVE system is suitable for applications where the user and the verifier do not necessarily trust each other. Furthermore, the proposed system appears to be sufficiently efficient compared to the existing scheme and can be used in real-life applications.

VIII. ACKNOWLEDGMENTS

This work has been performed by the BEAT project 7th Framework Research Programme of the European Union (EU), grant agreement number: 284989. The authors would like to thank the EU for the financial support and the partners within the consortium for a fruitful collaboration. For more information about the BEAT consortium please visit <http://www.beat-eu.org>.

REFERENCES

- [1] A. K. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 14, pp. 4–20, 2004.
- [2] S. Prabhakar, S. Pankanti, and A. K. Jain, "Biometric recognition: Security and privacy concerns," *IEEE Security & Privacy*, vol. 1, no. 2, pp. 33–42, 2003.
- [3] A. K. Jain, K. Nandakumar, and A. Nagar, "Biometric template security," *EURASIP J. Adv. Signal Process.*, vol. 2008, pp. 113:1–113:17, Jan. 2008.
- [4] N. K. Ratha, J. H. Connell, and R. M. Bolle, "Enhancing security and privacy in biometrics-based authentication systems," *IBM Systems Journal*, vol. 40, no. 3, pp. 614–634, 2001.
- [5] C. Roberts, "Biometric attack vectors and defences," *Computers & Security*, vol. 26, no. 1, pp. 14–25, 2007.
- [6] T. A. M. Kevenaar, G. J. Schrijen, M. van der Veen, A. H. M. Akkermans, and F. Zuo, "Face recognition with renewable and privacy preserving binary templates," in *Proceedings of the Fourth IEEE Workshop on Automatic Identification Advanced Technologies*, ser. AUTOID '05, 2005, pp. 21–26.
- [7] H. Feistel, "Cryptography and computer privacy," vol. 228, no. 5, pp. 15–23, May 1973.
- [8] F. Hao, R. Anderson, and J. Daugman, "Combining crypto with biometrics effectively," *IEEE Trans. Comput.*, vol. 55, no. 9, pp. 1081–1088, Sep. 2006.
- [9] S. Kanade, D. Petrovska-Delacrutz, and B. Dorizzi, "Cancelable iris biometrics and using error correcting codes to reduce variability in biometric data," in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, 20–25 June 2009, Miami, Florida, USA. IEEE, 2009, pp. 120–127.
- [10] A. Juels and M. Wattenberg, "A fuzzy commitment scheme," ACM Press, 1999, pp. 28–36.
- [11] G. I. Davida, Y. Frankel, and B. J. Matt, "On enabling secure applications through off-line biometric identification," in *IEEE Symposium on Security and Privacy*, 1998, pp. 148–157.
- [12] S. Tulyakov, F. Farooq, and V. Govindaraju, "Symmetric hash functions for fingerprint minutiae," in *ICAPR (2)*, 2005, pp. 30–38.
- [13] Y. Sutcu, Q. Li, and N. Memon, "How to protect biometric templates," in *SPIE Conf on Security, Steganography and Watermarking of Multimedia Contents IX*, 2007.
- [14] M. v. d. V. A. Stoianov A., T. Kevenaar, "Security issues of biometric encryption," in *Science and Technology for Humanity (TIC-STH)*, 2009 IEEE Toronto International Conference. IEEE, 2009, pp. 34–39.
- [15] X. Zhou, "Privacy and security assessment of biometric template protection," *it - Information Technology*, vol. 54, no. 4, pp. 197–, 2012.
- [16] X. Zhou, A. Kuijper, R. Veldhuis, and C. Busch, "Quantifying privacy and security of biometric fuzzy commitment," in *Proceedings of the 2011 International Joint Conference on Biometrics*, ser. IJCB '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 1–8. [Online]. Available: <http://dx.doi.org/10.1109/IJCB.2011.6117543>
- [17] S. Cimato, M. Gamassi, V. Piuri, R. Sassi, and F. Scotti, "A biometric verification system addressing privacy concerns," in *CIS*, 2007, pp. 594–598.
- [18] T. Matsumoto, H. Matsumoto, K. Yamada, and S. Hoshino, "Impact of artificial "gummy" fingers on fingerprint systems," *Datenschutz und Datensicherheit*, vol. 26, no. 8, 2002.
- [19] T. van der Putte and J. Keuning, "Biometrical fingerprint recognition: Don't get your fingers burned," in *CARDIS*, 2000, pp. 289–306.
- [20] J. Bringer, H. Chabanne, G. D. Cohen, B. Kindarji, and G. Zémor, "Optimal iris fuzzy sketches," *CoRR*, vol. abs/0705.3740, 2007.
- [21] M. Barni, T. Bianchi, D. Catalano, M. Di Raimondo, R. Donida Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, F. Scotti, and A. Piva, "Privacy-preserving fingercode authentication," in *Proceedings of the 12th ACM workshop on Multimedia and security*, 2010, pp. 231–240.
- [22] K. Nandakumar, A. K. Jain, and S. Pankanti, "Fingerprint-based fuzzy vault: Implementation and performance," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 4, pp. 744–757, 2007.
- [23] Y. C. Feng, P. C. Yuen, and A. K. Jain, "A hybrid approach for generating secure and discriminating face template," *Trans. Info. For. Sec.*, vol. 5, no. 1, pp. 103–117, march 2010.
- [24] F. M. Bui, K. Martin, H. Lu, K. N. Plataniotis, and D. Hatzinakos, "Fuzzy key binding strategies based on quantization index modulation (qim) for biometric encryption (be) applications," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 1, pp. 118–132, 2010.
- [25] A. Juels and M. Sudan, "A fuzzy vault scheme," *Des. Codes Cryptography*, vol. 38, no. 2, pp. 237–257, 2006.
- [26] C. Karabat and H. Erdogan, "A cancelable biometric hashing for secure biometric verification system," in *Proceedings of the 2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, 2009, pp. 1082–1085.
- [27] Z. Bai and D. Hatzinakos, "Lbp-based biometric hashing scheme for human authentication," in *Control Automation Robotics Vision (ICARCV)*, 2010 11th International Conference on, dec. 2010, pp. 1842–1847.
- [28] Y. W. Kuan, A. B. J. Teoh, and D. C. L. Ngo, "Secure hashing of dynamic hand signatures using wavelet-fourier compression with biophase mixing and 2n discretization," *EURASIP J. Adv. Sig. Proc.*, vol. 2007, 2007.
- [29] C. Rathgeb and A. Uhl, "Iris-biometric hash generation for biometric database indexing," *Pattern Recognition, International Conference on*, pp. 2848–2851, 2010.
- [30] R. Luminari and L. Nanni, "An improved biohashing for human authentication," *Pattern Recognition*, vol. 40, pp. 1057–1065, 2006.
- [31] W. J. Scheirer and T. E. Boulton, "Cracking fuzzy vaults and biometric encryption," in *Proceedings of Biometrics Symposium*, 2007, pp. 1–6.
- [32] A. Adler, "Vulnerabilities in biometric encryption systems," in *In International Conference on Audio and Video based Biometric Person Authentication*, 2005, pp. 1100–1109.
- [33] T. E. Boulton, W. J. Scheirer, and R. Woodworth, "Revocable fingerprint biotokens: Accuracy and security analysis," in *CVPR*, 2007.
- [34] A. Kong, K.-H. Cheung, D. Zhang, M. Kamel, and J. You, "An analysis of biohashing and its variants," *Pattern Recogn.*, vol. 39, pp. 1359–1368, Jul. 2006.
- [35] K. Kommel and C. Vielhauer, "Reverse-engineer methods on a biometric hash algorithm for dynamic handwriting," in *Proceedings of the 12th ACM workshop on Multimedia and security*, ser. MMSEC '10, 2010, pp. 67–72.
- [36] K. H. Cheung, A. W.-K. Kong, J. You, and D. Zhang, "An analysis on invertibility of cancelable biometrics based on biohashing," in *CISST*, 2005, pp. 40–45.
- [37] K. Kummel, C. Vielhauer, T. Scheidat, D. Franke, and J. Dittmann, "Handwriting biometric hash attack: a genetic algorithm with user interaction for raw data reconstruction," in *Proceedings of the 11th IFIP TC 6/TC 11 international conference on Communications and Multimedia Security*, 2010, pp. 178–190.
- [38] K. Simoens, P. Tuyls, and B. Preneel, "Privacy weaknesses in biometric sketches," in *Proceedings of the 2009 30th IEEE Symposium on Security and Privacy*, ser. IEEE SP'09, 2009, pp. 188–203.
- [39] T. Ignatenko and F. M. J. Willems, "Information leakage in fuzzy commitment schemes," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 337–348, 2010.

- [40] K. Simoens, J. Bringer, H. Chabanne, and S. Seys, "A framework for analyzing template security and privacy in biometric authentication systems," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 833–841, 2012.
- [41] T. Ignatenko and F. M. J. Willems, "Biometric systems: privacy and secrecy aspects," *Trans. Info. For. Sec.*, vol. 4, no. 4, pp. 956–973, Dec. 2009.
- [42] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, "Privacy-preserving face recognition," in *Proceedings of the 9th International Symposium on Privacy Enhancing Technologies*, ser. PETS '09, 2009, pp. 235–253.
- [43] A.-R. Sadeghi, T. Schneider, and I. Wehrenberg, "Efficient privacy-preserving face recognition," in *ICISC*, 2009, pp. 229–244.
- [44] M. Barni, T. Bianchi, D. Catalano, M. Di Raimondo, R. Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, A. Piva, and F. Scotti, "A privacy-compliant fingerprint recognition system based on homomorphic encryption and fingercode templates," in *Biometrics: Theory Applications and Systems (BTAS), 2010 Fourth IEEE International Conference on*, Sept. 2010.
- [45] N. K. Ratha, J. H. Connell, and R. M. Bolle, "An analysis of minutiae matching strength," in *Proc. 3rd AVBPA*, 2001, pp. 223–228.
- [46] A. Adler, "Sample images can be independently restored from face recognition templates," in *Proc. Canadian Conference on Electrical and Computer Engineering (CCECE)*, vol. 2, 2003, pp. 1163–1166.
- [47] J. Galbally, C. McCool, J. Fierrez, and S. Marcel, "On the vulnerability of face verification systems to hill-climbing attacks," *Pattern Recognition*, vol. 43, pp. 1027–1038, 2010.
- [48] C. Rahtgeb and A. Uhl, "Attacking iris recognition: An efficient hill-climbing technique," in *Proc. ICPR*, 2010.
- [49] M. Martinez-Diaz, J. Fierrez, J. Galbally, and J. Ortega-Garcia, "An evaluation of indirect attacks and countermeasures in fingerprint verification systems," *Pattern Recognition Letters*, vol. 32, pp. 1643–1651, 2011.
- [50] S. P. Umut Uludag, Sharath Pankanti and A. K. Jain, "Biometric cryptosystems: Issues and challenges," in *Proceedings of the IEEE*, 2004, pp. 948–960.
- [51] C. Rathgeb and A. Uhl, "Statistical attack against iris-biometric fuzzy commitment schemes," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*, 2011, pp. 23–30.
- [52] T. Ignatenko and F. Willems, "Secret rate - privacy leakage in biometric systems," in *Proceedings of the 2009 IEEE international conference on Symposium on Information Theory - Volume 4*, ser. ISIT'09, 2009, pp. 2251–2255.
- [53] A. Stoianov, "Security of error correcting code for biometric encryption," in *Eighth Annual Conference on Privacy, Security and Trust, PST 2010, August 17-19, 2010, Ottawa, Ontario, Canada*. IEEE, 2010.
- [54] E.-C. Chang, R. Shen, and F. W. Teo, "Finding the original point set hidden among chaff," in *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, 2006, pp. 182–188.
- [55] A. Kholmatov and B. Yanikoglu, "Realization of correlation attack against the fuzzy vault scheme," 2008.
- [56] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," *SIAM J. Comput.*, vol. 38, no. 1, pp. 97–139, Mar. 2008.
- [57] Y. Dodis, B. Kanukurthi, J. Katz, L. Reyzin, and A. Smith, "Robust fuzzy extractors and authenticated key agreement from close secrets," *IEEE Transactions on Information Theory*, vol. 58, no. 9, pp. 6207–6222, 2012.
- [58] Y. Sutcu, L. Qiming, and N. Memon, "Design and analysis of fuzzy extractors for faces," in *Proceedings of SPIE Optics and Photonics in Global Homeland Security V and Biometric Technology for Human Identification VI*, vol. 7305, 2009.
- [59] T. S. Ong and A. B. J. Teoh, "Fuzzy key extraction from fingerprint biometrics based on dynamic quantization mechanism," in *Proceedings of the Third International Symposium on Information Assurance and Security*, 2007, pp. 71–76.
- [60] A. Arakala, J. Jeffers, and K. J. Horadam, "Fuzzy extractors for minutiae-based fingerprint authentication," in *ICB'07*, 2007, pp. 760–769.
- [61] X. Boyen, "Reusable cryptographic fuzzy extractors," in *Proceedings of the 11th ACM conference on Computer and communications security*, ser. CCS '04, 2004, pp. 82–91.
- [62] M. G. Qiming Li and E.-C. Chang, "Fuzzy extractors for asymmetric biometric representation," in *Proceedings of IEEE Workshop on Biometrics (In association with CVPR)*, 2008.
- [63] Y. Sutcu, H. T. Sencar, and N. Memon, "A secure biometric authentication scheme based on robust hashing," in *Proceedings of the 7th workshop on Multimedia and security*, 2005, pp. 111–116.
- [64] A. T. B. Jin, K.-A. Toh, and W. K. Yip, "2^{ns} discretisation of biophases in cancellable biometrics," in *ICB*, 2007, pp. 435–444.
- [65] B. Yang, C. Busch, P. Bours, and D. Gafurov, "Robust minutiae hash for fingerprint template protection," in *Media Forensics and Security*, 2010.
- [66] P. Tuyls, A. H. M. Akkermans, T. A. M. Kevenaar, G. J. Schrijen, A. M. Bazen, and R. N. J. Veldhuis, "Practical biometric authentication with template protection," in *AVBPA*, 2005, pp. 436–446.
- [67] F. Kerschbaum, M. J. Atallah, D. M'Raihi, and J. R. Rice, "Private fingerprint verification without local storage," in *ICBA*, 2004, pp. 387–394.
- [68] R. W. Hamming, "Error detecting and error correcting codes," *Bell System Technical Journal*, vol. 29, pp. 147–160, 1950.
- [69] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1991, pp. 586–591.
- [70] M. Osadchy, B. Pinkas, A. Jarrous, and B. Moskovich, "Scifi - a system for secure face identification," in *Security and Privacy (SP), 2010 IEEE Symposium on*, May 2010, pp. 239–254.
- [71] S. Rane, W. Sun, and A. Vetro, "Secure distortion computation among untrusting parties using homomorphic encryption," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*, Nov 2009, pp. 1485–1488.
- [72] J. Bringer, H. Chabanne, and A. Patey, "Shade: Secure hamming distance computation from oblivious transfer," in *Financial Cryptography Workshops*, 2013, pp. 164–176.
- [73] R. Kulkarni and A. Nambodiri, "Secure hamming distance based biometric authentication," in *Biometrics (ICB), 2013 International Conference on*, June 2013, pp. 1–6.
- [74] M. S. Kiraz, B. Schoenmakers, and J. Villegas, "Efficient committed oblivious transfer of bit strings," in *Information Security*, ser. Lecture Notes in Computer Science, vol. 4779. Springer Berlin Heidelberg, 2007, pp. 130–144.
- [75] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-dnf formulas on ciphertexts," in *Proceedings of the Second International Conference on Theory of Cryptography*, ser. TCC'05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 325–341.
- [76] T. El Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *Proceedings of CRYPTO 84 on Advances in cryptography*, 1985, pp. 10–18.
- [77] P. Paillier and D. Pointcheval, "Efficient public-key cryptosystems provably secure against active adversaries," in *Proc. of Asiacrypt'99, Lecture Notes in Computer Science*. Springer Verlag, 1999, pp. 165–179.
- [78] J. Katz and M. Yung, "Threshold cryptosystems based on factoring," vol. 2501, pp. 192–205, 2002.
- [79] S. Goldwasser and S. Micali, "Probabilistic encryption," *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, 1984.
- [80] R. Cramer, I. Damgård, and J. B. Nielsen, "Multiparty computation from threshold homomorphic encryption," in *Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques: Advances in Cryptology*, ser. EUROCRYPT '01, 2001, pp. 280–299.
- [81] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *VLDB'99, Proceedings of 25th International Conference on Very Large Data Bases, September 7-10, 1999, Edinburgh, Scotland, UK*, 1999, pp. 518–529.
- [82] D. C. L. Ngo, A. T. B. Jin, and A. Goh, "Biometric hash: high-confidence face recognition," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 16, no. 6, pp. 771–775, 2006.
- [83] O. Goldreich, *The Foundations of Cryptography - Volume 1, Basic Techniques*. Cambridge University Press, 2001.